Q1. What is the meaning of multiple inheritance?

**Answer:** Multiple inheritance is a feature in object-oriented programming languages that allows a class to inherit from more than one base class. In other words, a class can derive attributes and behaviors from multiple parent classes simultaneously.

When a class uses multiple inheritance, it inherits the members (attributes and methods) from all the base classes it derives from. This means that the derived class has access to all the public and protected members of each base class. It can override inherited methods, add new methods, and define new attributes.

Q2. What is the concept of delegation?

**Answer:** The concept of delegation in object-oriented programming refers to a design pattern where an object forwards or delegates a specific task or responsibility to another object. Instead of implementing the task itself, the delegating object assigns the responsibility to a different object, known as the delegate or deputy. The delegate then performs the task on behalf of the delegating object.

Q3. What is the concept of composition?  
**Answer:** The concept of composition in object-oriented programming refers to a design principle where complex objects are created by combining or composing simpler objects. Composition allows objects to be composed of other objects, forming a part-whole relationship.

In composition, an object is made up of one or more other objects, which are often referred to as component objects or parts. The composite object encapsulates these component objects and provides a higher-level interface to interact with them.

Q4. What are bound methods and how do we use them?

**Answer:** Bound methods are a concept related to object-oriented programming and refer to methods that are associated with specific instances of a class. A bound method is a callable object that is bound to an instance, allowing it to access the instance's attributes and operate on them.

In Python, when you define a method within a class, it is by default a bound method. When you create an instance of that class and call the method on the instance, the method automatically receives the instance as the first argument (often named self). This binding of the method to the instance is what makes it a bound method.

Q5. What is the purpose of pseudoprivate attributes?

**Answer:** Pseudoprivate attributes, also known as name mangling or name hiding, are a feature in some programming languages, including Python, that allow class attributes to be "hidden" or made less accessible from outside the class. In Python, pseudoprivate attributes are denoted by prefixing their names with double underscores (\_\_).